EV622186095

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

SYSTEMS AND METHODS FOR ENCODING RANDOMLY DISTRIBUTED FEATURES IN AN OBJECT

Inventor(s):
Darko Kirovski

ATTORNEY'S DOCKET NO. MS1-1934US

TECHNICAL FIELD

The systems and methods described herein generally relate to counterfeit-resistant and/or tamper-resistant labels, and more particularly, to utilizing randomly distributed features of an object (whether embedded or naturally inherent) to limit unauthorized attempts in counterfeiting and/or tampering with the label.

BACKGROUND OF THE INVENTION

Counterfeiting and tampering of labels cost product marketers and manufacturers billions of dollars each year in lost income and lost customers. With the proliferation of computer technology, generating labels that resemble the genuine item has become easier. For example, a scanner may be utilized to scan a high-resolution image of a genuine label which can then be reproduced repeatedly at a minimum cost. Also, coupons may be scanned, modified (e.g., to have a higher value), repeatedly printed, and redeemed.

Various technologies have been utilized to stop the flood of counterfeiting and tampering in the recent years. One way labels have been secured is by incorporation of bar codes. Bar codes are generally machine-readable code that is printed on a label. Using a bar code scanner, the label with a bar code may be quickly read and authenticated. One problem with current bar coded labels is that an identical label may be used on various items.

Another current solution is to have the scanned bar code examined against secure data stored in a database (e.g., a point of sale (POS) system). This solution, however, requires incorporation of up-to-date data from a marketer or

manufacturer. Such a solution requires timely and close cooperation of multiple entities. Also, such a solution limits its implementation flexibility and may not always be feasible.

These technologies, however, share a common disadvantage; namely, the labels scanned are physically identical for a given product. Accordingly, even though the manufacturing process for creating the legitimate labels may be highly sophisticated, it generally does not take a counterfeiter much time to determine a way to create fake pass-offs. And, once a label is successfully copied a single time, it may be repeatedly reproduced (e.g., by building a master copy that is replicated at low cost). Even if a label is black-listed in a database after a given number of uses, there is no guarantee that the labels that are scanned first are actually the genuine labels.

Accordingly, the current solutions fail to provide labels that are relatively hard to copy and inexpensive to produce.

SUMMARY OF THE INVENTION

The systems and methods described herein are directed at encoding randomly distributed features in an object. In one aspect, randomly distributed features in an authentication object are determined. Data representing the randomly distributed features is compressed and encoded with a signature. A label is created and includes the authentication object and the encoded data.

In another aspect, the data is compressed by determining a probability density function associated with the authentication object. Vectors associated with the randomly distributed attributes are determined based, at least in part, on the

probability density function. The vectors are encoded using an arithmetic coding algorithm.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 shows an example authentication object for use as part of a label, such as a certificate of authenticity.

Fig. 2 is a schematic diagram illustrating an example certificate of authenticity system and example procedures employed by the system for issuing and verifying a certificate of authenticity.

Fig. 3A is a schematic diagram of an example scanning system for capturing randomly distributed features of an authentication object associated with a certificate of authenticity.

Fig. 3B is a top view of the authentication object shown in Fig. 3A.

Fig. 4 is a flow diagram of an example process that may be used to create a certificate of authenticity.

Fig. 5 is a flow diagram of an example process that may be used to compress data that represents the randomly distributed attributes of an authentication object.

Figure 6 is a graphical representation of areas that correspond to four different regions in an example authentication object.

Figure 7 is a graphical representation of the nineteen different regions on an example authentication object.

Fig. 8 is a graph of an example of the probability density function for a square authentication object.

Figure 9 is a graphical representation of areas in an authentication object.

Fig. 10 is a graphical representation of an example of how an arithmetic coder encodes the string "*aba*".

Figure 11 is an example of an instance of an authentication object shown with nodes.

Figure 12 is a graphical representation of a certificate of authenticity designed for optimizing cost effectiveness.

Fig. 13 illustrates an example computing device which the described systems and methods can be either fully or partially implemented.

DETAILED DESCRIPTION

I. Introduction

The systems and methods described herein are directed at encoding information about the randomly distributed features of an object used in a label. Labels may include any type of identification means that are attached to or incorporated within an item. A label that is configured to be authenticated is referred herein as a certificate of authenticity. An object with randomly distributed features used in a certificate of authenticity is referred to herein as an authentication object. To enable self-authentication, a certificate of authenticity may include both the authentication object and the information about the randomly distributed features. A compression method may be used to increase the amount of information about the randomly distributed features that can be encoded and included in the certificate of authenticity. According to one example calculation, the cost of forging a certificate of authenticity is exponentially increased proportional to the improvement in compressing the information. This substantial

increase in forging cost results in a reliable certificate of authenticity that is relative cheap to manufacture but is difficult to falsify.

Fig. 1 shows an example authentication object 100 for use as part of a label, such as a certificate of authenticity. To be effectively used in a certificate of authenticity, authentication object 100 typically contains randomly distributed features that are unique and are hard to replicate. The example authentication object 100 shown in Fig. 1 is part of a fiber-based certificate of authenticity and contains fibers 110 that are embedded in the object in a random manner. Fibers 110 serve as the randomly distributed features of authentication object 100. Fibers 110 may be incorporated in authentication object 100 by any means. For example, fibers 100 may be sprayed onto authentication object 100. Fibers 100 may also be embedded into authentication object 100 during the manufacturing process. In one embodiment, fibers 110 are optical fibers capable of transmitting light between their endpoints. Thus, by shedding light on a certain region 120 of authentication object 100, endpoints of fibers 131-133 that have at least one end-point within the lit up region are illuminated.

In Fig. 1, authentication object 100 includes κ randomly distributed fibers. Authentication object 100 may be scanned at a resolution of L x L pixels. Each fiber has a fixed length of R. Although the example authentication object 100 in Fig. 1 contains fibers, it is to be understood that authentication objects with other randomly distributed features may also be used in a certificate of authenticity in a similar manner.

The randomly distributed features of authentication object 100 may be used in a certificate of authenticity to protect the proof of authenticity of an arbitrary object, such as a product. For example, certain hard-to-replicate data

about the randomly distributed features of the certificate of authenticity may be digitized, signed with the private key of the issuer, and the signature may be imprinted on the certificate of authenticity in a machine-readable form to validate that the produced instance is authentic. Each instance of the certificate of authenticity is associated with an object whose authenticity the issuer wants to vouch. In one embodiment, verification of authenticity is done by extracting the signed data (data about the randomly distributed features) using the public key of the issuer and verifying that the extracted data matches the data of the associated instance of the certificate of authenticity. In order to counterfeit protected objects, the adversary needs to either: (i) figure out the private key of the issuer, (ii) devise a manufacturing process that can exactly replicate an already signed instance of the certificate of authenticity, or (iii) misappropriate signed instances of the certificate of authenticity. From that perspective, the certificate of authenticity can be used to protect products whose value roughly does not exceed the cost of forging a single certificate of authenticity instance, including the accumulated development of a successful adversarial manufacturing process.

A goal of a certificate of authenticity system is to ensure the authenticity of products or certain information associated with a product. The set of applications is numerous and broad, ranging from software and media (e.g., DVD, CD) anti-piracy to unforgeable coupons and design of tamper-proof hardware. For example, creating a tamper-resistant chip would require coating its package with a certificate of authenticity. Before each usage, the integrity of the certificate of authenticity should be verified in order to verify authenticity of the protected silicon.

Below, example hardware platforms for inexpensive but efficient read-out of the randomly distributed features of a fiber-based certificate of authenticity will be discussed. The hardware platforms may include a barcode. Since the capacity of a barcode for low-cost readers is limited to about 3K bits, the message signed by the private key is limited to the same length. Also, since one of the goals of a certificate of authenticity system is to maximize the effort of the adversary who aims at forging a specific instance of the certificate of authenticity, the problem associated with storing in the fixed-length signed message as much as possible information about the unique and randomly distributed features of a fiber-based certificate of authenticity will be discussed. An example analytical model for a fiber-based certificate of authenticity will be provided. Then, the discussion below will also formalize the problem of compression of a point set, and show that optimal compression of fibers' positions in an instance of a certificate of authenticity is an NP-complete problem. In order to heuristically address this problem, an algorithm which significantly improves upon compression ratios of conventional compression methodologies will be provided.

II. Issuing and Verifying Certificate of Authenticity

Fig. 2 is a schematic diagram illustrating an example certificate of authenticity system 200 and example procedures employed by the system for issuing and verifying a certificate of authenticity. Certificate of authenticity system 200 includes certificate of authenticity 210, an issuer 230, and a verifier 250. As shown in Fig. 2, certificate of authenticity 210 may include the authentication object 100 in Fig. 1, a barcode 213, and text 215.

The information that needs to be protected on a certificate of authenticity includes: (*a*) the representation of the hard-to-replicate randomly distributed features of authentication object 100 and (*b*) an arbitrary associated textual data. Initially, the randomly distributed features of authentication object 100, such as locations of fibers, are scanned using a hardware device. Details on how this information is collected and represented will be discussed below in conjunction with Fig. 3.

For the purpose of discussion, assume that the resulting information f is a random string of n_F bits. Parameter n_F is fixed and equals $n_F = k * n_{RSA}, k \in N$, where n_{RSA} is the length of an RSA public-key (for example, $n_{RSA} = 1024$) and k is commonly set to $k \in [1,3]$. Given a fixed n_F, the digest f of data 231 representing the randomly distributed features of authentication object 100 may statistically maximize the distance between any two distinct certificate of authenticity instances. This goal translates directly to minimized likelihood of a false negative and false positive during the verification step.

The textual data t is an arbitrary string of characters which depends on the application (e.g., expiration date, manufacturer's warranty). The textual data is derived from text 215, which is printed on certificate of authenticity 210 as shown in Fig. 2.

The textual data may be hashed using a cryptographically secure hash algorithm 237, such as SHA1. The output of the hash function is denoted as a message t with n_T bits. Issuer 230 creates the message m that may be signed by RSA. For example, messages f and t are merged into a message m of length $n_M = n_F$ using a reversible operator $\otimes$ that ensures that each bit of m is dependent upon all bits from both f and t. This step may maximize the number of bits that

need to be manipulated in data 231 as well as text 215 to create a certain message m. An example of such an operator is symmetric encryption $m = t \dot{\otimes} f \equiv E_t(f)$ of f using t or certain subset of bits from t as a key. Message m is signed with an RSA signature 235 using the private-key 233 of the issuer 230. Each n_{RSA} bits of m are signed separately. The resulting signature s has $n_S = n_M = n_F$ bits. This message is encoded and printed as barcode 213 (such as barcodes that obey the PDF417 standard) onto certificate of authenticity 210.

The verification of certificate of authenticity 210 involves several steps. Verifier 250 initially scans the printed components: text 215 and barcode 213. Barcode 213 is decoded into the originally printed signature s. Text 215 is scanned and is hashed in order to create the message t. Note that generic optical character recognition (OCR) is not required for this task because the font used to print the text is known to the verifier 250 and optimized for improved OCR. For successful certificate of authenticity verification, text 215 and barcode 213 need to be read without errors; a task which is readily achievable with modern scanning technologies.

Verifier 250 performs the RSA signature verification 255 on s using issuer's public-key 253 and obtains the signed message m. Verifier 250 can then compute $f = m(\dot{\otimes})^{-1} t$. In the example of using encryption as $\dot{\otimes}$, this is achieved via decryption $f = E_t^{-1}(m)$. Next, verifier 250 scans data 251 of representing the randomly distributed features in authentication object 251 and creates their presentation f'. Verifier 250 compares f' to the extracted f. Verifier 250 needs to quantify the correlation between the two sets of data: the one attached to the certificate and the one used to create the signature on the certificate of authenticity. At decision block 259, if the level of similarity of the two sets of

data surpasses a certain threshold, verifier 250 announces that the certificate of authenticity 210 is authentic and vice versa.

Fig. 3A is a schematic diagram of an example scanning system 300 for capturing randomly distributed features of authentication object 310 associated with a certificate of authenticity. Scanning system 300 includes optical sensor 322 and light source 324. Optical sensor 322 is configured to scan authentication object 310 and may include a charged coupled device (CCD) matrix of a particular resolution. In one embodiment, optical sensor 322 has a resolution of 128 x 128 pixels. Light source 324 is configured to provide light of a particular wavelength to illuminate a region of authentication object 310. Light source 324 may include, for example, a light emitting diode (LED). As shown in Fig. 3A, one end of fiber 326 in authentication object 310 is illuminated by light source 324. The light is transmitted to the other end of fiber 326 and is sensed by optical sensor 322.

Fig. 3B is a top view of the authentication object 310 in Fig. 3A. In operation, the scanning system 300 divides authentication object 310 into regions, such as regions 311-314. As shown in Fig. 3B, light source 324 of scanning system 300 sheds light onto region 314 while regions 311-313 are isolated from light source 324. By illuminating region 314, the location of the endpoints in regions 311-313 of authentication object 310 can be determined by optical sensor 322. Thus, the read-out of the randomly distributed features in authentication object 310 includes four digital images that contain four different point-sets. Each point-set is associated with a particular region and is determined by illuminating that region.

It is conceivable that advancement in technology, such as nanotechnology, may enable an electronic device to decode the randomly distributed features from

a certificate of authenticity and create a light pattern that corresponds to these features. Such a device may be able to forge the certificate of authenticity. In one embodiment, scanning system 300 may be configured to prevent this method of forging by changing the wavelength (e.g. color) of the light used by light source 324. For example, the wavelength of the light may be randomly selected each time an authentication object is scanned by scanning system 300. Optical sensor 322 may be configured to detect the wavelength of the light emitted by the fibers in the authentication object and to determine whether that wavelength corresponds to the wavelength of the light emitted by light source 324. If the wavelengths of the emitted and detected light do not match, the certificate of authenticity is likely a forgery.

Fig. 4 is a flow diagram of an example process 400 that may be used to create a certificate of authenticity. At block 405, the authentication object in a certificate of authenticity is scanned. The authentication object may be scanned using scanning system 300 in Fig. 3A.

At block 410, data representing the randomly distributed attributes of the authentication object is determined. In a fiber-based authentication object, the data may include the positions of the endpoints of fibers that are illuminated, such as the endpoints shown in Fig. 3B.

At block 415, the data is compressing to enhance the security level of the certificate of authenticity. Data compression will be discussed in detail in conjunction with Fig. 5. Briefly stated, a path may be determined for compressing a portion of the data representing randomly distributed attributes in the authentication object.

At block 420, the compressed data is encoded. For example, the compressed data may be signed using private-key 233 in Fig. 2. At block 425, the encoded data is incorporated in the certificate of authenticity. For example, the encoded data may be printed onto the certificate of authenticity as a barcode, such as barcode 213 in Fig. 2.

Fig. 5 is a flow diagram of an example process 500 that may be used to compress data that represents the randomly distributed attributes of an authentication object. For the purpose of discussion, process 500 will be described in the context of a fiber-based certificate of authenticity. However, process 500 may be applied to any type of certificate of authenticity.

At block 505, a probability density function associated with the authentication object is determined. Probability density function will be discussed in Section III-A. An example probability density function is shown in Equation 11. A graphical presentation of the example probability density function is illustrated in Fig. 8. Briefly stated, the probability density function represents the likelihood that a unit of the randomly distributed attributes is found in a certain location of the authentication object. In the context of a fiber-based certificate of authenticity, the probability density function may represent the probability that a particular point in a region of the authentication object is illuminated. The probability density function may also be used to compute how many of the total fibers will be illuminated in a particular region.

At block 510, vectors associated with the randomly distributed attributes are determined. In the context of a fiber-based certificate of authenticity, point-to-point vectors are used and will be discussed in Section IV-A. In particular,

Equation 16 may be used to compute point-to-point vectors to represent the randomly distributed attributes in a fiber-based certificate of authenticity.

At block 515, the vectors are encoded using an arithmetic coding algorithm. Arithmetic coding algorithm will be discussed in Section IV-A. An example algorithm is shown in Table 2.

At block 520, a path for compressing a portion of the vectors within a fixed amount of data is determined. The method for computing the path is discussed in Section IV-B. The example path may be computed using Equation 20. At block 525, the path of the compressed data representing a portion of the randomly distributed attributes is returned.

III. Certificate of Authenticity Model

In this section, an analytical model of a fiber-based certificate of authenticity is discussed. Two features of a certificate of authenticity S are modeled. Given that a particular region S_i of the certificate of authenticity is illuminated, the probability density function that a particular point in $S - S_i$ is illuminated is computed. Also, given that K fibers are in S, the expected number of fibers that are illuminated in $S - S_i$ is also computed.

A. Distribution of Illuminated Fiber End-Points

An authentication object (L,R,K) is modeled as a square with an edge of L units and K fibers of fixed length $R \leq L/2$ randomly thrown over the object. Other model variants, such as variable fiber length or arbitrary shape authentication object, can be derived from this model. The authentication object is positioned in the positive quadrant of a 2D Cartesian coordinate system as illustrated in Fig. 1.

In addition, the authentication object is divided into four equal squares $S=\{S_1,S_2,S_3,S_4\}$. Each of them is used to record the 3D fiber structure as described above in conjunction with Fig. 3A and 3B. Next, a fiber is denoted as a tuple $f=\{A,B\}$ of points $A,B\subset S$ such that the distance between them is $\|A-B\|=R$.

Definition 1. Distribution of Illuminated Fiber End-Points. Given that one of the squares S_i is illuminated, the probability density function (pdf) $\varphi(i,Q(x,y))$ is defined for any point $Q(x,y)\subset S-S_i$ via the probability $\xi(i,P)$ that any area $P\subset S-S_i$ contains an illuminated end-point A of a fiber $f=\{A,B\}$, conditioned on the fact that the other end-point B is located in the illuminated region S_i. More formally, for any $P\subset S-S_i$:

$$\xi(i,P)=\Pr[A\subset P\mid f=\{A,B\}\subset S,B\subset S_i] \quad (6)$$

$$=\underbrace{\iint}_{Q(x,y)\subset P}\varphi(i,Q(x,y))\mathrm{d}x\mathrm{d}y.$$

Assume that throwing a fiber $f=\{A,B\}$ into an authentication object consists of two dependent events: (*i*) first end-point A lands on the authentication object and (*ii*) second end-point B hits the authentication object. While A can land anywhere on the COA, the position of B is dependent upon the location of A. Endpoint B must land on part of the perimeter of the circle centered around A, with a radius R, and contained within the authentication object. In the remainder of this subsection, the function $\varphi(i,Q(x,y))$ is analytically computed based on the

analysis of the events (*i-ii*). For brevity, only $\varphi(1, Q(x,y))$ is computed for the case when region S_1 is lit up. $\varphi(1, Q(x,y))$ are computed in two steps.

Definition 2. Perimeter Containment. First, for a given point $A \subset S$, the perimeter containment function $\varrho(A)$ is defined, which measures the length of the part of the perimeter (arc) of the circle centered at A with radius R that is encompassed by the entire authentication object S. There are four different regions in the authentication object (marked P1 through P4 in Fig. 6) where $\varrho(A)$ is uniformly computed.

Fig. 6 is a graphical representation of areas P1-P4 that correspond to the four different regions in an example authentication object 600. For each point in a certain area PX, the perimeter containment function is computed using a closed analytical form distinct for that area using Equations 7-10 as discussed below.

AREA P1. This is the central area of the authentication object, where for any point $Q \subset$ P1, the circle with radius R centered at Q does not intersect with any of the edges of the authentication object. The area is bounded by: $R \le x \le L - R$, $R \le y \le L - R$.

$$\varrho(Q(x,y)) = 2R\pi. \qquad (7)$$

AREA P2. There are four different P2 regions, where a circle with radius R centered at any point $Q \subset$ P2 intersects twice with exactly one edge of the authentication object. For brevity, consideration is give only for the following one: $R \le x \le L - R$, $0 \le y < R$. Equations for other three regions can be symmetrically computed.

$$\varrho(Q(x,y)) = R\left[\pi + 2\arcsin\left(\frac{y}{R}\right)\right]. \quad (8)$$

AREA P3. There are four different P3 regions, where a circle with radius R centered at any point $Q \subset \text{P3}$ intersects twice with two different edges of the authentication object. Consideration is give only for the following one: $0 \le x < R$, $0 \le y < R$, $x^2 + y^2 \ge R^2$.

$$\varrho(Q(x,y)) = 2R\left[\pi - \arccos\left(\frac{x}{R}\right) - \arccos\left(\frac{y}{R}\right)\right]. \quad (9)$$

AREA P4. There are four different P4 regions, where a circle with radius R centered at any point $Q \subset \text{P4}$ intersects once with two edges of the COA. Consideration is give only for the following one: $x^2 + y^2 < R^2$.

$$\varrho(Q(x,y)) = R\left[\frac{\pi}{2} + \arcsin\left(\frac{x}{R}\right) + \arcsin\left(\frac{y}{R}\right)\right]. \quad (10)$$

In all Equations 8-10, only the return values of functions arcsin(·) and arccos(·) that are within $\{0, \pi/2\}$ are considered.

In the second step, the actual $\varphi(1, Q(x,y))$ is computed based on the fact that an illuminated endpoint A of a fiber $f = \{A, B\}$ is at position $A = Q(x,y)$ only if B is located on the part(s) of the circle $C(Q,R)$ centered at $Q(x,y)$ with a diameter R and contained by S_1.

Lemma 3. Dependence of $\varphi(i,Q(x,y))$ from $\varrho(Q(x,y))$. Using function $\varrho(Q(x,y))$, pdf $\varphi(i,Q(x,y))$ is computed using the following integral:

$$\varphi(i,Q(x,y)) = \int_{C(Q,R)\subset S_i} \frac{\alpha R \mathrm{d}\vartheta}{\varrho(Q(x+R\cos\vartheta, y+R\sin\vartheta))}. \qquad (11)$$

where ϑ browses the perimeter of $C(Q,R)\subset S_i$ and α is a constant such that:

$$\underbrace{\iint}_{Q(x,y)\subset S-S_i} \varphi(i,Q(x,y))\mathrm{d}x\mathrm{d}y = 1. \qquad (12)$$

A point $Q\subset S-S_i$ can be illuminated only due to a fiber $f=\{Q,B\}$, such that $B\subset S_i$. This implicates that B is located somewhere on the perimeter of the circle $C(Q,R)$ contained by S_i. For a given fiber $f=\{A,B\}$, the probability that A lands on a specific infinitesimally small arc of length $\mathrm{d}l\subset S$, is equal to $\mathrm{d}l/\varrho(B)$. Hence:

$$\varphi(i,Q) = \mathrm{area}(S-S_i)^{-1} \int_{C(Q,R)\subset S_i} \frac{4R\mathrm{d}l\mathrm{d}\vartheta}{\varrho(B(Q,R,\vartheta)\subset C)\mathrm{d}l}, \qquad (13)$$

where function $\mathrm{area}(S-S_i)$ computes the area under $S-S_i$. Thus, the pdf $\varphi(1,Q(x,y))$ at a point $Q\subset S-S_1$ is proportional to the integral of the inverse of the value of $\varrho(\cdot)$ over $C(Q,R)\subset S_1$.

Figure 7 is a graphical representation of the nineteen different regions on an example authentication object 700 that have distinct analytical formulae as a solution to the integral quantified in Equation 11. For brevity, $\varphi(1,Q(x,y))$ is

approximately solved using a simple numerical computation. The results is illustrated in Fig. 8

Fig. 8 is a graph of an example probability density function for a square authentication object with parameters $L = 64$ and $R = 28$ sampled at unit points. Fig. 8 shows that the likelihood that an endpoint of a fiber lands on a certain small area $P \subset S - S_1$ varies significantly depending on the particular position of P within $S - S_1$. By using the information about the variance of $\varphi(i, Q(x,y))$ throughout $S - S_i$, the point-subset compression algorithms can be significantly improved, as presented in Section IV. Manufacturing authentication object such that $\varphi(i, Q(x,y)) = const.$ over the entire area $S - S_i$, is a non-trivial task, probably as difficult as forging an original authentication object.

Area	Bounds	$\psi(1, Q(x,y))$
T0	$0 \le x \le L/2 - R,\ 0 \le y \le L/2 - R$	0
T1	$x^2 + (y - L/2)^2 < R^2,\ 0 \le x \le L/2 - R,$ $L/2 - R < y \le L/2$	$R\left[\arcsin\left(\frac{x}{R}\right) + \arccos\left(\frac{L/2-y}{R}\right)\right]$
T2	$x^2 + (y - L/2)^2 \ge R^2,\ 0 \le x \le L/2 - R,$ $L/2 - R < y \le L/2$	$2R\arccos\left(\frac{L/2-y}{R}\right)$
T3	$x^2 + (y - L/2)^2 \ge R^2,$ $(x - L/2)^2 + y^2 \ge R^2,$ $(x - L/2)^2 + (y - L/2)^2 \ge R^2$	$2R\left[\arccos\left(\frac{L/2-y}{R}\right) + \arccos\left(\frac{L/2-x}{R}\right)\right]$
T4	$x^2 + (y - L/2)^2 < R^2,$ $(x - L/2)^2 + y^2 < R^2,$ $(x - L/2)^2 + (y - L/2)^2 \ge R^2$	$R\left[\arcsin\left(\frac{x}{R}\right) + \arcsin\left(\frac{y}{R}\right)\right]$ $R\left[\arccos\left(\frac{L/2-y}{R}\right) + \arccos\left(\frac{L/2-x}{R}\right)\right] +$
T5	$x^2 + (y - L/2)^2 < R^2,$ $(x - L/2)^2 + y^2 < R^2,$ $(x - L/2)^2 + (y - L/2)^2 < R^2$	$R\left[\frac{\pi}{2} + \arcsin\left(\frac{x}{R}\right) + \arcsin\left(\frac{y}{R}\right)\right]$
T6	$x^2 + (y - L/2)^2 < R^2,$ $(x - L/2)^2 + y^2 \ge R^2,$	$R\left[\arcsin\left(\frac{x}{R}\right) + \arccos\left(\frac{L/2-y}{R}\right)\right] +$ $2R\arccos\left(\frac{L/2-x}{R}\right)$

	$(x-L/2)^2+(y-L/2)^2 \geq R^2$, $L/2-R<x\leq L/2$	
T7	$x^2+(y-L/2)^2<R^2$, $(x-L/2)^2+y^2\geq R^2$, $(x-L/2)^2+(y-L/2)^2<R^2$	$R\left[\frac{\pi}{2}+\arcsin\left(\frac{x}{R}\right)+\arccos\left(\frac{L/2-x}{R}\right)\right]$
T8	$x^2+(y-L/2)^2\geq R^2$, $(x-L/2)^2+y^2\geq R^2$, $(x-L/2)^2+(y-L/2)^2<R^2$	$R\left[\frac{\pi}{2}+\arccos\left(\frac{L/2-y}{R}\right)+\arccos\left(\frac{L/2-x}{R}\right)\right]$

Table 1.

B. Illumination Ratio of Fiber End-Points

Definition 3. Illumination Ratio of Fiber End-Points. For an authentication object (L,R,K) and its illuminated region S_i, the illumination ratio λ is defined as a probability that a fiber $f=\{A,B\}$ has landed such that one of its end-points is in $B\subset S-S_i$ conditioned on the fact that the other end-point is in $A\subset S_i$:

$$\lambda=\Pr[B\subset S-S_i \mid f=\{A,B\}, A\subset S_i]. \qquad (14)$$

Definition 4. Possibly Illuminated Arc. For any point $A\subset S_i$, a function $\psi(i,A(x,y))$ is defined that measures the length of the part of the perimeter of $C(A,R)$ contained by $S-S_i$.

Figure 9 is a graphical representation of the areas T0-T8, where $\psi(i,Q(x,y))$ is computed using distinct closed analytical forms. $\psi(i,Q(x,y))$ is analytically computed based on the analysis of the events (*i-ii*) from Section III-A. Similarly to Section III-A, only in the case when region S_i is lit up is computed. There are nine different regions in the COA (marked T0 through T8 in Fig. 9)

where $\psi(1,Q)$ is computed uniformly. The analytical closed forms for $\psi(1,Q)$ depending on the location of Q within S_1 are given in Table 1.

Lemma 4. Dependence of $\psi(1,Q(x,y))$, $\varrho(Q(x,y))$, and λ. The illumination ratio defined as in Def.3, can be computed as follows:

$$\lambda = \int_{Q(x,y)\subset S_i} \frac{\psi(i,Q(x,y))}{\varrho(Q(x,y))} \mathrm{d}x\mathrm{d}y. \qquad (15)$$

A circle centered at a point $A \subset S$ with radius R is denoted as $C(A,R)$. For each point $Q \subset S_i$, the likelihood that the other end-point B of a fiber $f = \{Q,B\}$ lands within $S - S_i$, equals the ratio of lengths of parts of the perimeter of $C(Q,R)$ contained by $S - S_i$ and S respectively. By integrating this ratio over all points within S_i, Equation 15 is obtained.

Given an authentication object (L,R,K), using λ, computed by numerically approximating Equation 15 and the closed forms for $\psi(1,Q)$ from Table 1, one can compute the expected number of illuminated points in $S - S_1$ when S_1 is illuminated as $\lambda K/2$. For example, for an authentication object (64,28,100) the resulting $\lambda \approx 0.74$, which means that on the average, the number of illuminated endpoints in case S_i is illuminated, is about $0.74 \cdot 50 = 37$.

IV. Compression of a Point-Subset in a COA

The goal of the certificate of authenticity system is to ensure that the task of manufacturing (i.e. forging) a specific authentication object instance as difficult as possible. This goal is quantified as a demand for recording the positions of as many as possible fibers of the authentication object. In the example compression

algorithm, the number of regions of authentication object equals four; hence, for each region S_i, a quarter $n_M/4$ of bits in the signed message m is dedicated to storing as many as possible fiber end-points illuminated in $S - S_i$ once light is shed on S_i. Note that in general, not all illuminated points need to be stored; only the largest subset of these points that can be encoded using $n_M/4$ bits.

In this section, a mechanism is described, which is configured to encode the distance between two illuminated points in an authentication object. The mechanism is based on arithmetic coding. Next, the problem of compressing as many as possible fiber endpoints using a constant number of bits is formalized. Finally, the discussion will show that this problem is NP-complete and a constructive heuristic as a sub-optimal solution is presented.

A. Encoding Point-to-Point Vectors

In this subsection, how a vector defined by its starting and ending point is encoded using a near-minimal number of bits is described. An additional constraint is that the points in the considered area occur according to a given pdf.

1) Arithmetic coding:

An arithmetic coder (AC) converts an input stream of arbitrary length into a single rational number within $[0,1\}$. The principal strength of AC is that it can compress arbitrarily close to the entropy. The discussion below shows how a word "*aba*" is encoded given an alphabet with an unknown pdf of symbol occurrence.

Fig. 10 is a graphical representation of an example of how an arithmetic coder encodes the string "*aba*" is encoded given an alphabet $L = \{a,b\}$ with an unknown pdf of symbol occurrence. The example is illustrated in Fig. 10.

Initially, the range of the AC is reset to $[0,1\}$ and each symbol in L is given an equal likelihood of occurrence $\Pr[a]=\Pr[b]=1/2$. Thus, the AC divides its range into two subranges $[0,0.5\}$ and $[0.5,1\}$, each representing "b" and "a" respectively. Symbol a is encoded by constraining the range of the AC to the range that corresponds to this symbol, i.e., $[0.5,1\}$. In addition, the AC updates the counter for the occurrence of symbol "a" and recomputes $\Pr[a]=2/3$ and $\Pr[b]=1/3$. In the next iteration, according to the updated $\Pr[a],\Pr[b]$, the AC divides its range into $[0.5,0.6667\}$ and $[0.6667,1\}$, each representing "b" and "a" respectively. When "b" arrives next, the AC reduces its range to the corresponding $[0.5,0.6667\}$, updates $\Pr[a]=\Pr[b]=2/4$, and divides the new range into $[0.5,0.5833\}$ and $[0.5833,0.6667\}$, each representing "b" and "a" respectively. Since the final symbol is "a", the AC encodes this symbol by choosing any number within $[0.5833,0.6667\}$ as an output. By choosing a number which encodes with the fewest number of bits (digits in our example), 0.6, the AC creates its final output. The decoder understands the message length either explicitly in the header of the compressed message or via a special *"end-of-file"* symbol.

The AC iteratively reduces its operating range up to a point when its range is such that the leading digit of the high and low bound are equal. Then, the leading digit can be transmitted. This process, called *renormalization*, enables compression of files of any length on limited precision arithmetic units. Performance improvements of classic AC focus on: using precomputed approximations of arithmetic calculations, replacing division and multiplication with shifting and addition.

An AC encodes a sequence of incoming symbols $s=s_1,s_2,...$ using a number of bits equal to source's entropy, $H(s)=-\sum_{s_i}\Pr[s_i]\log_2(\Pr[s_i])$. Hence, for

a semi-infinite stream of independent and identically distributed symbols, on a computer with infinite precision arithmetic, the AC is an optimal, entropy coder.

2. Arithmetic Encoding of a Min-Distance Point-to-Point Vector

Given an authentication object (L,R,K), it is assumed that light is shed on one of its quadrants, S_i. Next, we assume that the authentication object is partitioned into a grid of $L \times L$ unit squares $U = u(i,j), i = 1...L, j = 1...L$, where each $u(i,j)$ covers the square area within $x \in \{i-1, i], y \in \{j-1, j]$. Unit areas model the pixels of the digital scan of an authentication object. The resolution of the scan equals $L \times L$. Next, a principal point of a unit $u(x,y)$ is defined as a point Q_u with coordinates (x,y).

Lemma 5. Unit Illumination Likelihood. Assuming there are κ fibers with exactly one end-point in $S - S_i$, the probability that any unit area $u(x,y) \subset S - S_i$ contains at least one illuminated fiber end-point equals:

$$\tau(u) = \Pr[(\exists f = \{A,B\} \in F) A \subset u, B \subset S_i] \quad (16)$$
$$= 1 - \left[1 - \xi(i,u)\right]^{\kappa}.$$

And

$$\tau(u) = \Pr[(\exists f = \{A,B\} \in F) A \subset u, B \subset S_i] = 1 - \Pr[(\neg \exists c \in F) A \subset u, B \subset S_i] = 1 - \left(1 - \Pr[A \subset u, B \subset S_i \mid f = \{A,B\}]\right)^{\kappa}$$

From Equation 7, Equation 16 is concluded. In Section III-B, the expectation for κ is $\mathrm{E}[\kappa] = \lambda K/2$ is computed.

Problem 1. Dual Vector Encoding for COA. Conditioned on the fact that unit $u \subset S - S_i$ contains an illuminated fiber end-point, a goal is to encode using as few as possible bits the locations of two other illuminated units v_1 and v_2 relative to unit u. An additional constraint is that among all illuminated units in $S - S_i$, the principal points of v_1 and v_2, Q_1 and Q_2 respectively, are located at two shortest distances in Euclidean sense from the principal point of u, Q_u. A priority rule is set so that if a set of units $V, |V| > 1$ are at the same distance with respect to u, the one with the highest likelihood of illumination: $\text{argmax}_{v \subset V}(\tau(v))$ is encoded first.

Set U as a list of all unit areas in $S - S_i - u$.
List of all marked units, $M(u)$, is set to $M(u) = \varnothing$.
do
  Find all unit areas $V = \text{argmin}_{v \subset U} \| Q_v - Q_u \|$.
  do
    Find unit area $w = \text{argmax}_{v \in V} \xi(1, v)$.
    Set AC range for w to $\gamma(w,u)$ (see Eqns.17,18).
    Set of nodes ordered before w is $M_w(u) = M(u)$.
    $M(u) = M(u) \cup w$, $V = V - w$, $U = U - w$.
  while $V \neq \varnothing$
while $U \neq \varnothing$

Table 2. ALGORITHM A1.

The encoding of a unit-to-unit vector is done using an AC, which uses algorithm A1 to assign a corresponding range on the encoding interval for each encoding symbol, i.e. each unit $v \subset S - S_i$ different from the source unit u. For

each unit v, algorithm A1 assigns a range equal to the probability that v is one of the two closest illuminated units with respect to the source unit u. This probability is denoted as $p(v|u)$. In the case when $\kappa \gg 1$ units are expected to illuminate in $S - S_i$, $p(v|u)$ can be computed as follows:

$$p(v|u) = \tau(v) \prod_{w \subset M_v(u)} [1 - \tau(w)] + \qquad (17)$$

$$\sum_{w \subset M_v(u)} \tau(v)\tau(w) \prod_{z \subset M_v(u), z \neq w} [1 - \tau(z)],$$

where the set of units $M_v(u)$ is computed as in algorithm A1. For each unit v, algorithm A1 assigns a range $\gamma(v,u)$ used by the AC to encode v conditioned on the fact that u has already been encoded. This range is equal to:

$$\gamma(v,u) = \frac{p(v|u)}{\sum_{w \subset S - S_i} p(w|u)}. \qquad (18)$$

Thus, the two nearest illuminated units are encoded by construction near-optimally (e.g. the encoding is optimal on a processor with infinite precision arithmetic) because a sequence of symbols is encoded using a number of bits approximately equal to the entropy of the source:

$$H(u) = -\sum_{v \subset S - S_i} \gamma(v,u) \log_2 [\gamma(v,u)]. \quad (19)$$

Dual vector encoding is used as a primitive to encode a subset of points in the overall compression algorithm presented in the Section IV-B. Although the

encoding algorithm is near-optimal for the set of assumptions presented in Section IV-A.2, the same set of constraints is not valid for the overall compression goal, hence, the inherent optimality of using arithmetic coding with range allocation via A1 is discussed in Section IV-B.

B. Compression of a Point-Subset

The optimization problem of compressing the positions of as many as possible illuminated unit areas using a fixed number of bits is modeled. Consider the following directed complete graph with weighted edges. For each illuminated unit $u \subset S - S_i$, a node n_u is created. A directed edge $e(u,v)$ from node n_u to node n_v is weighted with the optimal length of the codeword that encodes the vector that points to v, $\omega(e(u,v)) = -\log_2[\gamma(v,u)]$ as in Equation 19, conditioned on the fact that u is already encoded. Lets denote this graph as $G(N,E,\Omega)$, where N, E, and Ω represent the set of nodes, directed edges, and corresponding weights respectively.

Problem 2. Compression of a Point-Subset (CPS).

INSTANCE: Directed, complete, and weighted graph $G(N,E)$ with a non-negative vertex function $\Omega: E \to R$, positive integer $l_{min} \in Z^+$, positive real number $\Lambda \in R^+$.

QUESTION: Is there a subset of $l > l_{min}$ nodes $N^* \subset N$ with a path through them, i.e. a permutation $< n^*_{\pi(1)}, \ldots, n^*_{\pi(l)} >$, such that the sum of weights along the path is:

$$\sum_{i=1}^{l-1} \omega(e(n^*_{\pi(i)}, n^*_{\pi(i+1)})) < \Lambda. \qquad (20)$$

Problem 2 models the optimization problem of compressing as many as possible (i.e. l) fiber end-points in an authentication object using a fixed storage (i.e. Λ). This problem is NP-complete as it can be shown that the ASYMMETRIC TRAVELING SALESMAN PROBLEM, ATSP, can be reduced to CPS, ATSP $\leq_m^p$ CPS, via binary search for Λ. In the remainder of this section, an efficient constructive heuristic A2 is presented that aims at solving this problem. The premier design requirement for the heuristic is fast run-time performance because each certificate of authenticity must be signed separately at a manufacturing line.

First, the distance measure between two nodes in N does not obey the triangle inequality for all nodes. Intuitively, the encoding procedure from Section IV-A encodes vectors in $S - S_i$ using a number of bits proportional to the likelihood that a certain unit is one of the two closest illuminated points. Hence, units farther from the source node are encoded with significantly longer codewords as they are unlikely to occur, which renders shortcuts to these nodes in the solution route highly undesirable.

Theorem 2. The distance measure ω does not universally obey the triangle inequality:

$$\omega(e(u,v)) + \omega(e(v,w) \geq \omega(u,w).$$

For simplicity, assume that $(\forall u \subset S - S_i)\, t = \tau(u) = \text{const.}$, then u, v, and w are positioned along the same line in $S - S_i$. The Euclidean distances $\|u - v\|$, $\|v - w\|$, and $\|u - w\|$ are a, b, and $a+b$ respectively. The triangle inequality implies that $f(u,v,w) = \log_2[\gamma(w,u)] - \log_2[\gamma(v,u)] - \log_2[\gamma(w,v)] \geq 0$. From Equations 17 and 18, the following can be computed:

$$f(a,b,t) = 2ab\pi \log_2(1-t) + \log_2 \frac{t}{1-t} - \\ -\log_2 \frac{(1-t)^2 + (a^2+b^2)\pi t(1-t) + a^4 b^4 \pi^2 t^2}{1 + \left[(a+b)^2 \pi - 1\right] t}, \tag{21}$$

and show that for $ab\pi t \gg 1$, the triangle inequality does not hold, i.e., $f(a,b,t) < 0$.

The best approximation algorithm for ATSP where the triangle inequality holds, yields solutions at most $log(|N|)$ times worse than the optimal. Alternatively, to the best knowledge of the authors, approximation algorithms for ATSP variants where the triangle inequality does not hold, have not been developed. In the general case, when the distance metric function ω is arbitrary, the ATSP problem is NPO-complete, i.e. there is no good approximation algorithm unless $P = NP$. On the other hand, approximation algorithms for variants of TSP which satisfy a scaled version of the triangle inequality: $\mu(\omega(e(u,v)) + \omega(e(v,w))) \geq \omega(u,w), \mu > 1$ can be solved with a worst case result $(3\mu+1)\mu/2$ times worse than the optimal solution. Distance metric ω does not follow this constraint, hence, a heuristic for Problem 2 is developed without a worst-case guarantee. In addition, we aim for as good as possible performance of the heuristic on the average, rather than a worst-case guarantee. Authentication

object instance which cannot be compressed satisfactorily can be disposed. Likelihood of this event should be small, less than one in a million.

CONSTRUCTIVE PHASE

Set of edges $E' = \{\text{argmin}_e(\omega(a,b),\omega(b,a)) \mid (\forall a,b) \subset N\}$.

Set of subpaths P is selected as a set of shortest K edges in E' s.t. $\sum_{i=1}^{K} \omega(e_i) \le \Lambda$ sorted by ω.

Denote the weight of the shortest edge in E as $\omega_{\min}$.

for each path $p_i \subset P, i = 1..K-1$

for each path $p_j \subset P, j = i+1..K$

if p_i and p_j have a common source-destination node

Concatenate p_i and p_j as $p_i = p_i \mid p_j$.

Remove p_j from P.

Denote source and destination nodes of a path $p_i \subset P$ as s_i and d_i respectively.

for each path $p_i \subset P, i = 1..K$

Find all shortest paths $q(i,j)$ from s_i to any $d_j, j \neq i$.

while $|P| <$ maxP

$(p_i, p_j) = \text{argmin}_{q(i,j)} \sum_{e \subset \{p_i \mid q(i,j) \mid p_j\}} \frac{\omega(e)}{|\{p_i \mid q(i,j) \mid p_j\}|}$.

Concatenate $p_i = p_i \mid q(i,j) \mid p_j$ and remove p_j from P.

Find exhaustively a concatenation $p_h = p_1 \mid ... \mid p_{\text{maxP}}$ s.t. $M(p_h)\{\sum_{e \subset p_h} \omega(e) < \Lambda$ and $|p_h|$ is maximal$\}$.

reroute(p_h)

reroute(p_h)

$p_{\text{best}} = p_h$

for each edge $e(s_i, d_i) \subset p_h, i = 1,...,|p_h|-1$

for each node pair $(d_i,\ s_j) \subset p_h, j = i+2,...,|p_h|-1$.

Find shortest path $q(i,j)$ via nodes in $N - p_h$.

if path $e_1,...,e_i \,|\, q(i,j) \,|\, e_j,...,e_{|p_h|}$ has a better metric $M(p_h)$ then p_{best}

then $p_{\text{best}} = p_h$.

GREEDY ITERATIVE IMPROVEMENT

repeat I times

Contract p_h so that $\sum_{e \subset p_h} \omega(e) \le \rho\Lambda$, where ρ is a contraction factor, randomly chosen from $\rho \in \{0.4, 0.8\}$.

Denote nodes n_0 and n_l as the first and last node in p_h.

while $\sum_{e \subset p_h} \omega(e) \le \Lambda$

Among edges that have n_0 or n_l as destination or source respectively, find edge e with minimal weight.

Concatenate e to p_h.

rereoute(p_h)

TABLE 3. ALGORITHM A2.

The rationale behind using the distance metric ω from Section IV-A is based on an assumption that a good solution succeeds to traverse each node on its route via the two closest neighboring nodes. Hence, in the scope of Problem 2, the used metric is optimal only if the best solution found satisfies this property. If the final solution does not have this property, the optimality of encoding a single vector is dependent upon the distribution of weights of the edges in the solution.

The developed heuristic A2 has two stages: a constructive and an iterative improvement phase. The constructive phase follows a greedy heuristic which builds the initial solution. Initially, A2 identifies a set of dominating edges E'. For each pair of edges, $e(u,v)$, $e(v,u)$, between nodes u,v, A2 selects only the shorter of the two and stores it in E'. Next, a set P of initial subpaths is created by sorting the edges in E' and selecting the top K shortest edges whose weights sum up as close as possible to Λ. The first and last node in a path p_i are denoted as s_i and d_i respectively. In the next step, A2 concatenates subpaths from P iteratively in the increasing order of their weights: at any point, the pair of shortest subpaths p_i, p_j which have a common source-destination node $d_i = s_j$, is concatenated until all possible connections are established. In the unlikely case when $|P|=1$, the optimal solution is found and the search is stopped. Else, all single-edge subpaths are removed from P. Then, using Dijkstra's algorithm, A2 finds all shortest paths between each destination tail d_i of each subpath p_i in P and source tails of all other subpaths, $s_j, i=1...|P|, i \neq j$. The shortest paths are routed via nodes which are not in P. The shortest path is denoted between s_i and d_j as $q(i,j)$. In another greedy step, A2 sorts all concatenations $p_i | q(i,j) | p_j$ according to their weight/node count ratio. In increasing order of this metric, A2 continues concatenating subpaths in P via nodes in $N-P$ until the total number of remaining paths is $|P|=\text{maxP}$ (usually $\text{maxP}=9$). The remaining paths are concatenated using an exact algorithm which finds a path p_h with the optimal metric: maximal cardinality and a sum of weights smaller than Λ. In the final step, a rerouting procedure browses all the nodes in P, and using the Dijkstra algorithm tries to find shortest paths to other nodes in P via the remaining nodes in E. The same procedure also tries to find a better ending tail than the one that exists in p_h.

For each reroute, A2 checks whether the new reroute has a better metric than the current, best path p_h.

Figure 11 is an example of an instance of an authentication object (512,0.4·512,256) is shown with $\kappa = 88$ nodes. A2 returned the path illustrated with bold lines. The path is such that its sum of weights is smaller than $\Lambda = 512$. To document the path, 12.11 bits per point is used.

In the iterative improvement phase, we repeat several rounds of the following loop. In the first step, A2 contracts the currently best found path p_{best} into p_h, so that $|p_h|$ is maximal and the sum of weights along p_h is smaller than a fraction of $\rho\Lambda$. The contraction parameter ρ is randomly selected in each iteration within $\rho \in \{0.4, 0.8\}$. Nodes n_0 and n_l are denoted as the first and last node in p_h. While the sum of weights in p_h is smaller than Λ, among edges that have n_0 or n_l as destination or source respectively, we find an edge e with minimal weight and concatenate it to p_h. When the new candidate path p_h is created, it is adopted as the best solution if its metric is better than the metric of the best path created so far. As a last step of the iterative improvement loop, A2 performs the rerouting procedure previously described.

In order to fit the run-time of A2 for a particular authentication object (L,R,K) class within one second, the improvement loop is repeated $I = \{100, 10000\}$ times. In general, the worst-time complexity of A2 is $O(|N|^3 \log |N|)$ as multi-source shortest paths are computed via the Dijkstra algorithm. In an implementation that uses the Floyd-Warshall algorithm to compute all pairs shortest paths, the complexity of A2 can be reduced to $O(|N|^3)$. Although the graph is originally complete, by removing edges with high weights, we create a

sparse graph, where Johnson's algorithm for all-pairs shortest paths yields $O(|N|^2 \log|N|+|N||E|)$.

V. Empirical Evaluation

The discussion in this section shows how authentication object (L,R,K) parameters impact the performance of the algorithm A.2. Figure 11 illustrates a solution to a single instance of the problem, an authentication object $(512, 0.4 \cdot 512, 256)$. The scanning grid to $L = 512$ scanning cells. The figure depicts the case when the lower left quadrant of the authentication object is illuminated. Graph $G(N,E)$, built using the corresponding illuminated fiber end-points, is illustrated with medium bold lines. Only the top ten shortest edges starting from each of the $\kappa = 88$ nodes in the graph is shown. The resulting path shown in the figure using bold lines, consists of 41 nodes. The sum of weights along path's edges is smaller than the storage limit: $\Lambda = 512$ bits. The path is compressed using 12.11 bits per fiber end-point (b/fep). Storing the data without compression would require $41 \cdot 18 = 738$ bits, which results in a compression ratio of 0.61. The compression ratio is defined as a ratio of the size of the compressed message vs. the original message size.

VI. A Design Objective for a COA System

A goal of the certificate of authenticity designer is to maximize the cost of forgery ς_f using a bounded manufacturing cost ς_m. Several parameters may impact ς_m. For brevity and simplicity, three parameters are discussed:

the total length of fiber $RK \le \Phi$,

the scanning tolerance ζ, and

the barcode storage Λ.

System performance is optimized by limiting the number of trials available to the adversary for accurate positioning of a sufficient subset of the signed fiber end-points (Section VI-A) and by selecting the system parameters $\{R_*, K_*\}$ so that expected forging cost $\varsigma_f(A2)$ is maximized (Section VI-B).

A. Limiting the Number of Adversarial Trials

Consider a compression scheme C which stores G out of the κ illuminated fiber end-points in a Λ-limited storage. In general, when forging a certificate of authenticity, the adversary can use all κ fibers to try to place at least $G\zeta$ of them accurately at their corresponding locations. Cost of forging a certificate of authenticity greatly depends upon the number of available trials. Here, a technique is proposed which aims at reducing the number of adversarial trials, K_T, by detecting anomalous distribution of fibers around the signed fiber end-points during verification.

ISSUING A COA INSTANCE

Scan for a set N of κ points, illuminated when light is shed on S_i.

Using Λ bits, compress a subset $P \subset N$, with $G = |P| \leq \kappa$.

Find a subset of units $U \subset S - S_i$, such that

$$(\forall u_i \in U)(\forall p_j \in P)\min(\| u_i - p_j \|) < \varepsilon_1 .$$

$\varepsilon_2 = |N \cap U| - G$, $K_T = G + \varepsilon_2$.

Sign P, ε_2 and the associated information (see Section 2).

VERIFYING A COA INSTANCE

Extract P, ε_2 from signature.

Find a subset of units $U \subset S - S_i$, such that

$(\forall u_i \in U)(\forall p_j \in P) \min(\| u_i - p_j \|) < \varepsilon_1$.

Scan for a set N' of κ' points, illuminated when light is shed on S_i.

if $| N' \cap U | > K_T$ **then** COA instance is invalid,

elseif $| N' \cap P | \geq G\zeta$ **then** COA instance is valid,

else COA instance is invalid.

TABLE 4. ALGORITHM A3

The certificate of authenticity issuer and verifier repeat their parts of the algorithm A3 for each authentication object quadrant S_i. The issuer initially scans the authentication object instance and collects information about the set of points N which illuminate when S_i is lit up. Next, using the available Λ bits, it compresses the largest subset $P \subset N$, $| P | = G$ returned by A2. Then, A3 finds a subset $U \subset S - S_i$, such that the Euclidean distance between each unit $u_i \in U$ and its closest unit $p_j \in P$ is at most ε_1. Subset U of units represents an ε_1-neighborhood of P. Then, the issuer counts the number K_T of points in N that exist in U. Since, K_T has to be greater than G to prevent false negatives, the issuer stores along with P, the difference $\varepsilon_2 = K_T - G$ in the message m, which is later signed using the private key of the issuer (see Section II). Using the public key of the issuer, the verifier extracts from the attached signature the compressed point subset P and ε_2 and recreates the corresponding ε_1-neighborhood, U. Then, the verifier scans the authentication object instance for the set of illuminated fibers N' when S_i is lit up. It announces that the instance is authentic by checking that

the number of common points in U and N' is at most $G+\varepsilon_2$ and that the number of common points in N' and P is at least $G\zeta$.

By storing ε_2 in the signature, the adversary is imposed to use at most $K_T = G+\varepsilon_2$ trials that position fibers in the ε_1-neighborhood of P. The adversary's goal is to place at least $G\zeta$ fiber end-points from P accurately, hence, the adversary can afford $G(1-\zeta)+\varepsilon_2$ misplacements located in the ε_1-neighborhood of P during the forgery process. It is expected that each trial, targeting a point p_i, if unsuccessful, ends up in the ε_1-neighborhood of p_i. By increasing ε_1, the verifier can identify possible misplacements over a larger neighborhood; however, this also increases the expectation for ε_2 - a value that the certificate of authenticity designer wants to keep as low as possible.

Below, an empirical design methodology is shown which adopts a given $\varepsilon_1 = \text{const.}$, and then seeks to maximize the main objective $\varsigma_f(A2)$ from the perspective of several certificate of authenticity parameters.

B. Designing a COA System

Problem 3. A Design Objective for a COA System. For a given compression algorithm A2, fixed $RK \le \Phi$, ζ, ε_1, and Λ, find a cut $\{R_*, K_*\}$ of the available fiber which maximizes:

$$\{R_*, K_*\} = \arg \max_{\{R,K|RK\le\Phi\}} \varsigma_f(A2, R, K), \quad (22)$$

where ς_f is defined in Lemma 2. Note that the number of trials κ in Eqn.2 equals K_T as presented in Subsection VI-A. Compression performance G in Equation 2 depends upon the efficacy of A2.

Figure 12 is a graphical representation of a certificate of authenticity design for optimized cost effectiveness. The abscissa quantifies fiber length R relative to L, while the ordinate shows the number of fibers K. The bar illustrates the log-cost of forgery $\log_{10}(\varsigma_f(A2,R,K))$ with a constraint limit $\Lambda = 512$ bits and a set of fixed parameters: $\zeta = 0.9$, $\varepsilon_1 = 8$, and $\nu = 0.8$. The figure also illustrates the quality of solutions obtained for all cuts of a fixed length fiber $RK = \Phi = 100L$.

A simple empirical technique may be used that searches for the best fiber cut $\{R_*, K_*\}$. The search procedure is illustrated using Figure 12. The abscissa and the ordinate represent the values of R and K respectively. The bar denotes the expected log-cost of forging an certificate of authenticity instance, $\log_{10}(\varsigma_f(A2,RK))$. The cost is given with respect to R and K, and for a fixed set of parameters: $\Lambda = 512$, $\zeta = 0.9$, $\varepsilon_1 = 8$, and $\nu = 0.8$. The diagram in Figure 12 was computed empirically. A2 is applied to 500 randomly generated certificate of authenticity (512,R,K) instances with each combination of $R = \{0.05L, 0.10L, ..., 0.45L\}$ and $K = \{80, 96, ..., 192, 256, 384, 512, 768, 1024\}$. The expected compression performance for each point in the remaining portion of the $\{R,K\}$-space was obtained by interpolating the empirical results. From Figure 12, the best fiber cut can be found in the neighborhood of $K_* \approx 900$ and $R_* \approx 0.1L$. This result points to the fact that for the selected design environment, a cross-shaped certificate of authenticity is the best option. Note that careful selection of the fiber cut resulted in an order of magnitude improvement in the forgery cost with respect to a randomly selected point on $RK = \Phi$. The empirical principles used in this example, can be applied to search for a near-optimal parameter set for different certificate of authenticity environments and manufacturing constraints.

Fig. 13 illustrates an example computing device 1300 within which the described systems and methods can be either fully or partially implemented. Computing device 1300 is only one example of a computing system and is not intended to suggest any limitation as to the scope of the use or functionality of the invention.

Computing device 1300 can be implemented with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use include, but are not limited to, personal computers, server computers, thin clients, thick clients, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, gaming consoles, distributed computing environments that include any of the above systems or devices, and the like.

The components of computing device 1300 can include, but are not limited to, processor 1302 (e.g., any of microprocessors, controllers, and the like), system memory 1304, input devices 1306, output devices 1308, and network devices 1310.

Computing device 1300 typically includes a variety of computer-readable media. Such media can be any available media that is accessible by computing device 1300 and includes both volatile and non-volatile media, removable and non-removable media. System memory 1304 includes computer-readable media in the form of volatile memory, such as random access memory (RAM), and/or non-volatile memory, such as read only memory (ROM). A basic input/output system (BIOS), containing the basic routines that help to transfer information

between elements within computing device 1300, such as during start-up, is stored in system memory 1304. System memory 1304 typically contains data and/or program modules that are immediately accessible to and/or presently operated on by processor 1302.

System memory 1304 can also include other removable/non-removable, volatile/non-volatile computer storage media. By way of example, a hard disk drive may be included for reading from and writing to a non-removable, non-volatile magnetic media; a magnetic disk drive may be included for reading from and writing to a removable, non-volatile magnetic disk (e.g., a "floppy disk"); and an optical disk drive may be included for reading from and/or writing to a removable, non-volatile optical disk such as a CD-ROM, DVD, or any other type of optical media.

The disk drives and their associated computer-readable media provide non-volatile storage of computer-readable instructions, data structures, program modules, and other data for computing device 1300. It is to be appreciated that other types of computer-readable media which can store data that is accessible by computing device 1300, such as magnetic cassettes or other magnetic storage devices, flash memory cards, CD-ROM, digital versatile disks (DVD) or other optical storage, random access memories (RAM), read only memories (ROM), electrically erasable programmable read-only memory (EEPROM), and the like, can also be utilized to implement exemplary computing device 1300. Any number of program modules can be stored in system memory 1304, including by way of example, an operating system 1320, application programs 1328, and data 1332.

Computing device 1300 can include a variety of computer-readable media identified as communication media. Communication media typically embodies

computer-readable instructions, data structures, program modules, or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" refers to a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared, and other wireless media. Combinations of any of the above are also included within the scope of computer-readable media.

A user can enter commands and information into computing device 1300 via input devices 1306 such as a keyboard and a pointing device (e.g., a "mouse"). Other input devices 1306 may include a microphone, joystick, game pad, controller, satellite dish, serial port, scanner, touch screen, touch pads, key pads, and/or the like. Output devices 1308 may include a CRT monitor, LCD screen, speakers, printers, and the like.

Computing device 1300 may include network devices 1310 for connecting to computer networks, such as local area network (LAN), wide area network (WAN), and the like.

Although the invention has been described in language specific to structural features and/or methodological steps, it is to be understood that the invention defined in the appended claims is not necessarily limited to the specific features or steps described. Rather, the specific features and steps are disclosed as preferred forms of implementing the claimed invention.